Parallel Utilization of SQL and NoSQL Databases – A Systematic Literature Review

Nađa Klještanović, Sonja Ristić, Darko Stefanović
University of Novi Sad
Faculty of Technical Sciences
Department of Industrial Engineering and Engineering Management
Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia
{nadja.kljestanovic, sdristic, darko.stefanovic}@uns.ac.rs

Abstract. *Complex information systems often face challenges when choosing an appropriate database for storing business logic data. SQL and NoSQL systems represent standard approaches with clearly defined areas of application. However, systems that lie at the intersection of these approaches encounter difficulties, as it is often necessary to take advantage of both models. The modern trend of polyglot persistence enables combination of multiple data models within a single system. This paper conducts a systematic literature review of the concept of polyglot persistence, as well as multi-model databases, providing a clear overview of the advantages, limitations and use cases of these approaches in various business and technical environments.*

Keywords. SQL, NoSQL, ACID, BASE, polyglot persistence, multi-model databases

1 Introduction

Many applications nowadays are data-intensive. Data ecosystem evolves in several dimensions like volume, variety, complexity and velocity. Traditional relational database management systems (RDBMS) with their Structured Query Language (SQL) enable efficient data organization, access and manipulation. SQL allows users to retrieve, read and modify data in the database in a simple way, providing intricate query patterns (Majhadi & Machkour, 2021). In that way relational databases (often referred as SQL databases) are suitable for analytics and report systems where data integrity is paramount.

Although RDBMSs have many advantages they have several limitations. They are primarily aimed for processing structured data. Working with semi-structured and unstructured data requires more complex customization and that can slow down system efficiency. Besides, RDBMSs are designed for vertical scaling, which involves upgrading existing server resources to increase database capacity. In addition, working with these systems requires a defined database schema, which can be a challenge in dynamic environments with rapidly changing data structures. The adherence to ACID (atomicity, consistency, isolation, durability) properties in transaction management slows down performance when dealing with large amounts of data. These strict properties require complex synchronization mechanisms. Finally, providing high system availability is a challenge because traditional relational systems often require complex data replication and failure recovery strategies (Ohlsson & Persson, 2019).

An umbrella term *NoSQL databases* encompasses a variety of data systems that do not adhere to relational data model and have different approaches to data storage and retrieval. Although the first models of this type of databases were created in the 1960s, their wider utilization and popularization began only after 2007, when the company Amazon presented its NoSQL database DynamoDB. Compared to relational databases, NoSQL databases offer a lot of advantages. They provide high performance in terms of data processing speed and capacity during storing large amounts of data. These advantages often come with a certain compromise in relation to ACID properties. NoSQL databases very often adhere BASE (Basically Available, Soft State, Eventual Consistency) instead of ACID properties. (Ohlsson & Persson, 2019).

Although NoSQL databases offer many advantages, like scalability, flexibility and high performance, they also have some challenges (Nurhadi et al., 2024). Adhering BASE properties can lead to temporary data inconsistency and problems with integrity of stored data. Also, NoSQL databases are not designed for complex queries and data transactions like SQL databases. That can make difficult analysing data or working with relations between entities. In addition, NoSQL databases are often schema-less, unlike SQL databases. Because of that, determining an appropriate data structure requires careful system design and understanding. The deficiency of standardization

among different types of NoSQL databases can make transfer across data management systems or integration into existing data systems difficult.

The practice of using both NoSQL and SQL databases within a single application is increasingly common, likewise using one system providing characteristics of both database types. These approaches are applied especially in complex architectures that require a combination of flexibility and scalability provided by NoSQL and the structure and integrity of data ensured by means of SQL databases. First approach, known as polyglot persistence, involves the use of different types of databases. Second approach, known as the multi-model database approach, implies the existence of just one database with SQL and NoSQL properties. Both approaches enable organizations to optimize system performance according to specific requirements, thereby ensuring efficient handling of different types of data, scalability, flexibility and integrity (Glake et al., 2022).

The goal of this research is to conduct a systematic literature review (SLR) to analyse and compare different types of NoSQL databases in combination with the relational database, addressing the lack of comprehensive studies on how these combinations are applied in the same projects. The result is a systematic overview of the advantages and disadvantages of various SQL/NoSQL combinations with special emphasis on their use in several use cases within the reviewed papers.

Following Introduction, this paper proceeds as follows. The second chapter includes an overview of the research planning, the method of its implementation and the analysis of the selected papers using descriptive statistical analysis. The third chapter presents the discussion about the research with answers to the research questions, while the fourth and final chapter are based on presenting opinions on the topic and potential directions for future work.

2 Systematic Literature Review

According to Barbara Kitcheman (2004), SLR consists of three key phases which are planning, conducting the review and reporting the review.

As part of this study, scientific papers dealing with applications that simultaneously utilize SQL and NoSQL databases were analysed. Special emphasis was placed on the overview of database systems, as well as their applications across different domains.

2.1 Planning the review

At the beginning of the first phase of the research, a search for existing papers on the same or similar topic was conducted. Several SRLs were found, but they are different from this paper in several aspects. First, they focus just on the analysis of the concept of polyglot persistence or multi-model databases. Second, they analyze only one specific combination of SQL and NoSQL databases. Finally, they do not provide a comprehensive overview of the various concepts and possible ways of their utilization.

The next step in planning an SLR is defining its objective. The aim of this research is to compare use cases involving different combinations of SQL and NoSQL databases. A comparative analysis presents commonly used combinations of these database systems. Furthermore, the focus is placed on the specific purpose of each presented combination.

2.1.1 Research Questions

The following research questions were formulated from the previously defined objective:

RQ1: What are the characteristics, advantages and limitations of SQL and NoSQL databases?

RQ2: What are polyglot persistence and multi-model approaches and how are they applied in contemporary information systems?

- **RQ2.1**: What are the key benefits of polyglot persistence approach?
- **RQ2.2**: What are the advantages of multi-model approach?

RQ3: What combinations of SQL and NoSQL databases are commonly used?

- **RQ3.1**: In which scenarios is a specific combination of SQL and NoSQL databases utilized?

2.2 Conducting the review

Conducting the review refers to the implementation of research, which includes the identification of relevant studies, their selection and quality assessment, as well as data extraction and summarization of primary research results (Kitchenham, 2004). The conducted research steps are shown in Fig. 1.

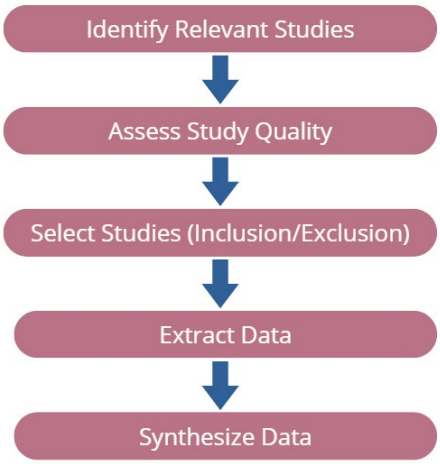

Figure 1. Conducted research steps

2.2.1 Identification of Research

Index databases used to search for relevant sources on the selected topic are Google Scholar and Scopus.

It was necessary to define the search string with different variations of key terms, while conditions related to the year of publication, type of work and language were set to maintain the relevance of the results. The search string for Scopus database is:

("SQL" OR "relational database" OR "relational databases") AND ("NoSQL" OR "non-relational database" OR "non-relational databases") AND ("polyglot persistence" OR "hybrid database" OR "hybrid architecture" OR "multi-model") AND PUBYEAR > 2016 AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar")) AND LIMIT-TO (LANGUAGE , "English").

Since Google Scholar uses a different search syntax compared to Scopus, the following text provides a search string relevant for this index database.
Additionally, it is necessary to apply filters for the publication year and document type after entering the string:

("SQL" OR "relational database" OR "relational databases") AND ("NoSQL" OR "non-relational database" OR "non-relational databases") AND "polyglot persistence" OR "hybrid database" OR "hybrid architecture" OR "multi-model").

2.2.2 Study Selection

Inclusion and exclusion criteria were defined based on the specific needs of the research. Criteria that were not applied during the search itself were implemented during the selection of works obtained through the search. A tabular overview of the inclusion and exclusion criteria is provided in Table 1 and Table 2, respectively.

Table 1. Inclusion criteria

ID	Description
I1	The paper must be written in English.
I2	The paper must be published in 2017 or later.
I3	Only journal articles or conference papers are accepted.
I4	The works must be accessible.
I5	The paper must describe use cases of parallel use of SQL and NoSQL databases in the same application.

Table 2. Exclusion criteria

ID	Description
E1	Duplicates must be deleted.
E2	If there are several works by the same author on the same topic, only one is taken.
E3	Papers dealing only with comparing the use of SQL or NoSQL databases in specific use cases are excluded.
E4	Papers discussing migration between SQL and NoSQL database systems are excluded.

During the further analysis of the selected works, the focus was placed on specific technologies and use cases involving the combination of NoSQL and SQL databases. Additionally, the descriptive statistical analysis considers all previously established criteria to provide a clearer statistical representation of the selected works.

2.2.3 Study Quality Assessment

Quality of study can be assessed like inclusion and exclusion criteria, but with a greater level of detail. This assessment provides that the studies are reliable for research. In order to ensure scientific quality, it is crucial to ask the following questions during evaluation process:

QA1: Were appropriate data collection methods used?
QA2: Are the research questions clearly formulated and relevant to the research area?
QA3: Are the results of the study reproducible under the same experimental conditions?
QA4: Are the used methodologies presented clearly?
QA5: Were the limitations of the study and their potential impact on the results considered?

The quality assessment was conducted on the studies that satisfied the defined inclusion and exclusion criteria. Only studies that fully met all quality assessment criteria were selected for the next phase of the research.

2.2.4 Data Extraction

The data extraction phase was conducted manually by the authors to provide a consistent and systematic analysis of studies, enabling answers to research questions and achieving quality criteria (Kitchenham, 2004). It involves a several groups of extracted information. The first group includes basic details about each study: ID, title, authors, year and place of publication. The second group of extracted data categorize studies according to the type of publication. The third data group classifies studies based on keywords, abstracts and titles. Finally, data related to the technologies used in the studies are extracted. This approach provides an objective analysis of the common utilization of these technologies.

On this basis, the grouping of data is done, which forms the foundation for the presenting data in the next phase, data synthesis.

2.2.5 Data Synthesis

The initial search of both selected index databases resulted in a total of 599 papers. Then the papers were selected based on defined inclusion and exclusion criteria, as well as based on the abstract, content, and the numbers of selected papers are shown in Table 3.

Table 3. Number of selected papers by search phases

	Scopus	Google Schoolar
Initial search	403	196
Number of selected papers based on inclusion, exclusion and quality assessment criteria	65	40
Number of selected papers based on abstract	30	22
Number of selected papers based on content	13	10
In total:	23	

Within this section, various types of data on the collected works are presented using tables and graphs. The presentation of basic information about the selected papers is based on their publication year, source type, content, relevance to the research questions, and the mentioned technologies. The information was extracted from 23 selected papers.

The first graph (Fig. 2) shows the percentage of published papers based on inclusion criteria. The years of publication of the papers range from 2017 to the present. As observed, the highest percentage of selected works was published in 2019 and 2022, accounting for 23.1% of all selected papers.

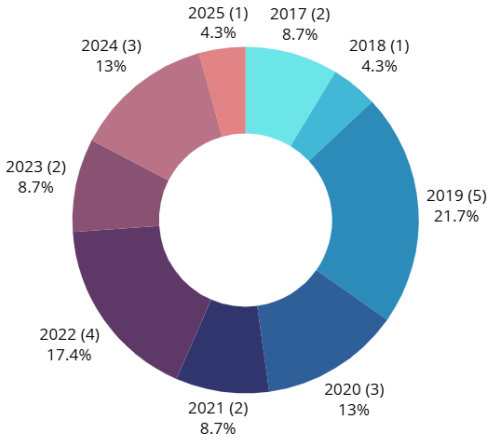

Figure 2. Publications pre year

Table 4 provides insight into the types of data sources, indicating whether the selected papers were published in conferences or scientific journals. The most of selected papers, nearly 77%, are classified as journal articles, while the remaining papers were published in conference proceedings.

Table 4. Source types of research

Source type	References	Percentage	Total
Conference proceeding	12, 14, 20, 21, 23	21.74%	5
Journal article	7, 8, 9, 10, 11, 13, 15, 16, 17, 18, 19, 22, 24, 25, 26, 27, 28, 29	78.26%	18

The subsequent division of selected papers is based on their content, that is, on the main topic of each research paper. Due to the diversity of topics, the papers are divided into three groups. The first group includes works dealing with migration between architectures, like migration from monolith to hybrid architecture as well as the reverse process. The second group consists of papers related to the integration of SQL and NoSQL databases, focusing on the integration of different database types through specific use cases. The third group refers to the papers that present theoretical explanation of polyglot persistence and multi-model databases. The papers are classified according to their primary topic and the results of this analysis are presented in the graph Fig. 3. It can be observed that the largest number of papers provides a theoretical introduction to the concepts of hybrid databases, which are the central focus of this research.

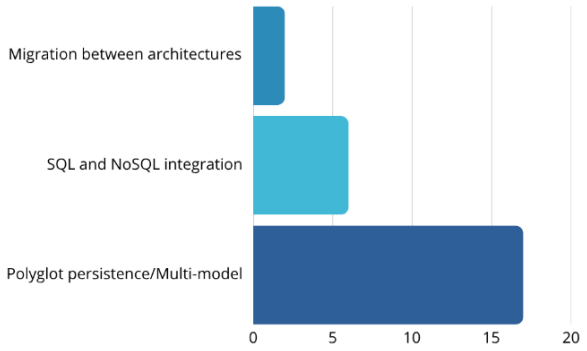

Figure 3. Research contexts

The following table presents the relevance of the selected papers to the research being conducted. Specifically, Table 5 provides an insight into which papers address specific research questions. The first column contains the serial number of the research question, which also refers to all sub-questions listed within that question. The second column lists the serial numbers of the papers that provide answers relevant to a particular research question.

Table 5. Answers to the research questions

Research question	References
RQ1	7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 21, 23, 24, 26, 27, 28, 29
RQ2	7, 9, 10, 11, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28
RQ3	7, 8, 10, 13, 15, 16, 17, 18, 20, 22, 24, 25, 26

The last graph (Fig. 4) presents the number of mentions of a particular technology within the selected works, as well as the context in which the technology appears. More precisely, specific types of databases may be mentioned independently as a tool, while others are referenced as part of a hybrid database. Graph presented in Fig. 4 displays the technologies that appear more than three times in the analyzed works, along with the number of occurrences for individual purposes, when the technology is part of the hybrid architecture, as well as the total number of occurrences in the papers. This approach can be useful for analyzing the frequency of technology utilization in research on polyglot persistence and multi-model concepts. The most often mentioned database in both approaches is MongoDB.

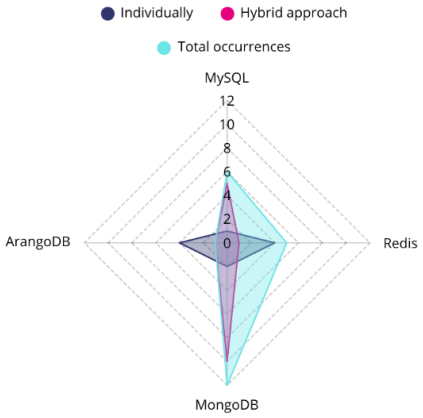

Figure 4. Number of technology occurrences

3 Discussion

The third phase of the SLR focuses on the results of the research. This part includes presenting the results in the form of a technical report. The aim of this paragraph is to answer the previously provided research questions.

RQ1 refers to the advantages and limitations of SQL and NoSQL databases. The important drawback of relational databases, mentioned by Lajam and Mohammed (2022), is the storage of exclusively structured data. Also, they have high costs of processing data and executing queries on tables due to joining tables and checking constraints. In research "*Recent trends in database technology*", (Lieponienė, 2020) concludes that relational databases prioritize data integrity, often at the expense of availability. Strict integrity enforcement prevents easy distribution of data. Data migrations are slower with these databases, but precisely because of the focus on data consistency and integrity, as stated by (Harezlak, Mermon, & Kasprowski, 2021).

In contrast, NoSQL databases have gained traction by addressing many of the limitations inherent in relational systems. Since data is kept in its original form, they offer improved performance and reduced programming costs. Through the study, (Lieponienė, 2020) it is concluded that the priority of NoSQL databases is data handling, not their security. They do not provide the possibility of built-in security mechanisms, but developers are responsible for their implementation in middleware. According to the research (Harezlak, Mermon, & Kasprowski, 2021), the main disadvantage of NoSQL databases is the absence of a standardized query language. Also, NoSQL databases cannot be optimized, which makes their use difficult, especially in a heterogeneous environment, as mentioned by (Kumar & Rampalli, 2019) in their research.

In (Zhang et al., 2022) autors discuss possible improvements within RDBMSs. Mainstream relational databases are working on optimizing their performance by adapting to big data through the JSON (Java ScripObject Notation) text storage format. In this way, SQL databases achieve efficiency by adopting NoSQL characteristics.

For that reason, the next research question (RQ2) logically focuses on the concept of polyglot persistence and multi-model databases, their characteristics and practical applications. Key differences between these two approaches are shown in Table 6.

To begin with, (Bimonte et al., 2023) discuss two possible approaches for integrating different types of data into one system. These are the parallel use of two or more database management systems (polyglot persistence) and the integration of all data into one DBMS (multi-model). Figure 5, as presented in (Mussbacher et al., 2024), clearly illustrates the hierarchy of multi-language development.

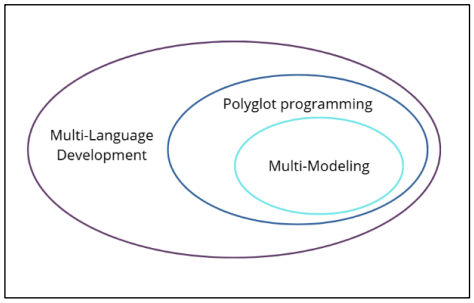

Figure 5. Multi-Language Development Hierarchy presented in (Mussbacher et al., 2024)

Table 6. Comparison of Polyglot Persistence and Multi-Model Databases

Characteristic	Polyglot persistence	Multi-model database
Definition	Multiple DBMSs	Single DBMS with multiple models
Architecture	Distributed	Unified
Complexity	High (integration, learning curve)	Low (single system)
Query language	Multiple	One (unified)
Scalability	Independent per DBMS	Centralized
Consistency	Hard to maintain	Easier
Cost	Higher (many tools, integration)	Lower (single platform)
Use cases	Heterogeneous data, e-commerce, analytics	Smart traffic, healthcare, Customer Relationship Management

Key characteristic of polyglot persistence is that an application uses more database systems of different types to solve conflicting requests. This allows different databases to complement each other and compensate for gaps. However, this architecture certainly increases programming complexity and requires knowledge of different types of databases. If implemented successfully, polyglot persistence brings a great advantage. This approach is presented from a competitive angle in (Peña, 2023). As the environment is dynamic and the company tends to react agilely and quickly to changes in business needs, properly implemented polyglot persistence can effectively support this adaptability. It optimizes data architecture, simplifies operations and enhances the efficiency and effectiveness of the decision-making process. Certainly, Lieponienė (2020) emphasizes that the implementation of this concept, including the choice of technologies, depends on how the stored data is utilized.

Polyglot data systems should keep important features of each database (like scalability), allow quick data changes using a global interface, automatica detection of changes between databases, run queries efficiently across systems, support real-time data processing, and multi-model schema management (Kiehn et al., 2022).

A good example that illustrates the concept of polyglot persistence is provided in (Lajam & Mohammed, 2022). The authors examine the functioning of e-commerce applications. Customer purchase data can be stored using a key-value NoSQL database.

However, if a client wants to know what their friends are buying, a graph database would be a more relevant option. Finally, for the execution of purchases and payments, a relational database is the most suitable choice. This is one example where polyglot persistence is a good choice to take advantage of different types of databases.

Polyglot persistence also has disadvantages. These relate to increased software complexity and a lack of support for maintaining data consistency. As databases are interconnected with mechanisms of other databases, it requires additional effort for their managing. The authors (Kazanavičius, Mažeika & Kalibatienė, 2022) notify that this concept is used only in cases where it is necessary to store different data models, otherwise there is a risk of overload. The disadvantages of polyglot persistence are the technical challenges of managing multiple databases, complex logic in applications, insufficient performance optimization, a steep learning curve for developers, and data consistency, according to (Bimonte et al., 2023). Using multiple databases makes data handling more complex due to differences in models, query languages, and systems, as noted by (Kumar & Rampalli, 2019). The hybrid database model produces different output formats, which directly increases complexity (Vyawahare et al., 2019). Recent studies focus on developing a hybrid framework that uses both databases to truly leverage the benefits of polyglot persistence.

Instead of dealing with the complexity of implementing the principle of polyglot persistence, there is the possibility of using multi-model databases. Multi-model databases simplify application maintenance and eliminate the need for different databases, as noted by (Subramanian & Saravanan, 2024). In their paper "*Holistic evaluation in multi-model databases benchmarking*", (Zhang & Lu, 2021) talk about multi-model databases as the next generation of DBMSs because they integrate flexibility, scalability and consistency. They predict that all leading data management systems will soon support multiple data models on a single platform.

The main advantage of the multi-model database is a unique system for all models, the maintenance of only one database, unique query language, architecture, adapter and topology, as stated by (Lieponienė, 2020). In addition, it is important to emphasize that the cost of scaling is lower because only one database is scaled, instead of several.

The authors of (Zhang & Lu, 2021) additionally note that multi-model databases also support agile development. However, these databases do not fully satisfy requirements such as scalability and performance. In cases where applications only need a limited set of data models, multi-model databases are recommended by (Kazanavičius, Mažeika & Kalibatienė, 2022).

Table 7. Commonly used technology combinations

SQL DB	NoSQL DB	Use Cases	Advantages	Ref.
MySQL	MongoDB, Redis	Real-time geospatial, IoT	Complex query optimization, real-time access, processing large amounts of data, reliable transactions, fast processing sensors data	8, 18, 22
	MongoDB	Web applications with user behavior tracking	Reliable transactions, flexibility with unstructured data (JSON format support)	20, 24
	HBase	High-volume data applications with transactions	High scalability, SQL query support over NoSQL	20
SQLite	MongoDB, Redis	E-commerce applications	Fast in-memory operations, easy integration, flexibility with unstructured data	16
Oracle	MongoDB		High security, support for large number of users, fast data insertion and updating, complex query optimization, data integration, data consistency	13, 17
PostgreSQL	Redis		Reliable transactions, availability-oriented	7
Any SQL	Cassandra	IoT systems (sensor + users and devices data)	Time-stamped data storage, processing large amounts of data	25

The authors (Zhang & Lu, 2021) point out that multi-model databases are useful in applications that operate in areas such as e-commerce, healthcare, online recommendation systems and smart traffic management.

A key part of the effectiveness of a polyglot system is the choice of the database, so RQ3 and its sub-questions are focused on frequently used technologies and their use cases. Table 7 present a summary of commonly used techology combinations in selected papers.

When asked which technology to choose for an application, one possible answer is often based on the most widely used databases.

The DB-Engines website has a table of database processing systems ranked by factors such as the number of mentions on websites, general interest in the system, number of active discussions, and number of job offers involving a given technology. According to the given criteria, Oracle (Oracle Corporation, 2025) relational and multi-model database is in the first place. Then, MySQL (MySQL, 2025), Microsoft SQL Server (Microsoft, 2025), and PostgreSQL (PostgreSQL Global Development Group, 2025) occupy the following positions. MongoDB (MongoDB, 2025) is the highest-ranked NoSQL database (DB-Engines, 2025). The technologies from the top of the ranking list are precisely those most frequently mentioned in selected papers.

Another way to select technologies, more complete than the previous one, involves matching the type of database with the purpose of the future application. In paper "*An integration approach of hybrid databases based on SQL in cloud computing environment*", (Li & Gu, 2019) deal with cloud services. In this area, hybrid databases are becoming a trend. NoSQL databases suitable for this architecture are Redis (Redis, 2025), MongoDB, HBase (Apache Software Foundation, Hbase documentation, 2025), Neo4j (Neo4j, 2025) and Memcached (Memcached, 2025). For MongoDB, HBase, and Neo4J systems, adding a SQL access layer is relatively easier because they have built-in Application Programming Interfaces (API) that support filtering and sorting records based on multiple conditions. In the case of multithreading and queries to multiple databases, MongoDB and Neo4j together perform better in execution time and resource usage, emphasized by (Ye et al., 2023). On the other hand, Redis and Memcached are characterized by difficult integration with SQL systems because they lack such APIs and require their construction first. A good practice when using a MongoDB in a hybrid architecture is to use it for user profile data or configuration data, while SQL databases handle the execution of e-commerce system transactions. The research by (Li & Gu, 2019) also points out the necessary use of SQL databases in banking systems for operational data, while HBase is used for analyzing large amounts of data. Redis and Memcached pair well with MySQL or PostgreSQL relational databases. Redis is presented by (Wu et al., 2017) as a tool suitable for high-traffic applications, i.e., those with many users accessing the system simultaneously. Cassandra (Apache Software Foundation, Hbase documentation, 2025) is a NoSQL database suitable for storing large volumes of time-stamped data, such as data collected from sensors. For this reason, it is often used in IoT (Internet of Things) systems in combination with SQL databases. In such architectures, SQL databases store data about users and devices.

A hybrid database composed of MySQL and MongoDB is discussed in (James & Asagba, 2017). The authors designed the system so that components can operate independently or together, providing flexibility in data management. The main task of MySQL is to handle transactions, while MongoDB plays a key role in data processing due to its efficient reading and writing characteristics.

According to a research evaluation conducted by (Li & Gu, 2019), the combination of MySQL, MongoDB, and Redis has competitive advantages in terms of optimizing complex queries, simplifying complex conditional clauses and certainly the types of

integrated data sources. This hybrid database is a good approach to solving the challenges of processing large amounts of real-time geospatial data. Also (Wu et al., 2017) in his work proposes the use of this combination of technologies.

A similar combination of technologies for applications dealing with e-commerce is presented in (Kazanavičius, Mažeika & Kalibatienė, 2022). A mentioned hybrid database is made up of Redis, MongoDB and SQLite (SQLite Consortium, 2025).

The combination of MySQL and HBase technologies is recommended by (Krishnapriya, Libin, & Gibin, 2021) in their work "*A study for integrating SQL and NoSQL Database*", when there is a need for high scalability to work with large amounts of data and process transactions. Using Apache Phoenix as a SQL translator for HBase allows SQL queries to be executed against a NoSQL database, facillitating integration with a MySQL database easier.

Also, (Krishnapriya, Libin, & Gibin, 2021) mention the MySQL and MongoDB hybrid database, which is well-known and frequently utilized in e-commerce web applications or for tracking user behavior. JSON is a commonly used format in web applications, and MongoDB is useful for storing data in that format.

Oracle is a relational database, suitable for a hybrid architecture alongside the MongoDB NoSQL database according to the research conducted by (Pokorný, 2019). This combination provides a higher level of business security when managing data and well supports systems with a large number of users. The authors (Bjeladinovic, Marjanovic, & Babarogic, 2020) also talk about this combination. They state that MongoDB enables faster data insertion and updating, especially when the data structure is variable. Oracle DBMS features exceptional join efficiency and high-performance multi-table query execution. Thus, the combination is characterized by good performance of complex queries, high data integrity and consistency.

(Bjeladinovic, 2025) in his research mentioned a previous database combination to which Cassandra was added. This improvement affects performance when working with large datasets, although that is not always the case with smaller amounts of data.

The last combination is mentioned by (Khine & Wang, 2019), which is a hybrid database consisting of PostgreSQL and Redis. The SQL part of this database is used for secure bank transactions in online shopping. NoSQL database Redis is responsible for implementing the user's cart due to its availability-oriented design.

At the end of discussion we would like to emphasize the main threat to validity. Namely, the presented analysis relies solely on data available in the Scopus and Google Scholar databases. In the future research more databases should be used to find relevant articles and in that way increase the number of selected articles. Besides, the usage of hybrid database system is evolving in time and we believe that the number of relevant articles will rise in further years.

4 Conclusion

The results of the conducted systematic literature review indicate a wide range of applications for the approaches of polyglot persistence and multi-model databases. Characteristics such as flexibility, scalability and data integrity represent the key requirements of modern information systems, and the analyzed approaches enable these goals to be achieved efficiently. The identified use cases and functionalities suggest significant potential for further development and wider adoption of these solutions in the future, despite the challenges associated with their implementation. Relational database management systems bring complex query optimization alongside with high data integrity and secure data transactions. On the other side, NoSQL databases can contribute to scalability, availability, flexibility with unstuctured data, and processing of large amounts of data. The diverse and fast-changing space of data systems technologies demands that designers and developers understand the concepts, pros and cons, and good practices of both of SQL and NoSQL worlds.

Future research should focus on proper implementation of polyglot persistence, in order to fully leverage its performance and improve the efficiency of business systems. Of particular importance is the development of recommendations for the optimal use of these concepts, tailored to the specific needs of users and the characteristics of information systems, which would represent a logical and valuable direction for further research in this area.

Acknowledgments

This research has been supported by the Ministry of Science, Technological Development and Innovation (Contract No. 451-03-137/2025-03/200156) and the Faculty of Technical Sciences, University of Novi Sad through project "Scientific and Artistic Research Work of Researchers in Teaching and Associate Positions at the Faculty of Technical Sciences, University of Novi Sad 2025" (No. 01-50/295).

References

Majhadi, K., & Machkour, M. (2021). The history and recent advances of Natural Language Interfaces for Database Querying. *E3S Web of Conferences (ICCSRE'2020)*, 229, 01039. https://doi.org/10.1051/e3sconf/202122901039

Ohlsson, A., & Persson, M. (2019). A Coparison in Performance Between a Selection of Databases (Using the benchmark tool YCSB). *Axis Communications AB* (Master's thesis).

Glake, D., Kiehn, F., Schmidt, M., Panse, F., & Ritter, N. (2022). Towards Polyglot Data Stores. *arXiv:2204.05779v1 [cs.DB]*. https://arxiv.org/abs/2204.05779

Nurhadi, R., Kadir, R. A., Mat Surin, E., & Sarker, M. R. (2024). A Systematic Review of Automated Classification for Simple and Complex Query SQL on NoSQL Database. *Computer Systems Science and Engineering, 48(6)*, 1434*, Tech Science Press.* https://doi.org/10.32604/csse.2024.051851

DB-Engines. (225). *Complete Ranking https://db-engines.com/en/ranking*

Khine, P. P., & Wang, Z. (2019). A Review of Polyglot Persistence in the Big Data World. *Information, 10*(4), 141. https://doi.org/10.3390/info10040141

Kitchenham, B. (2004). *Procedures for performing systematic reviews*. Keele University Technical Report TR/SE-0401, ISSN: 1353-7776, & NICTA Technical Report 0400011T.1. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=29890a936639862f45cb9a987dd599dce9759bf5

Li, C., & Gu, J. (2019). An integration approach of hybrid databases based on SQL in cloud computing environment. *Software: Practice and Experience*, 49, 401–422. https://doi.org/10.1002/spe.2666

Lajam, O., & Mohammed, S. (2022). Revisiting polyglot persistence: From principles to practice. *International Journal of Advanced Computer Science and Applications, 13*(5). https://doi.org/10.14569/IJACSA.2022.0130599

Kumar, K. P., & Rampalli, G. (2019). An efficient design for multiple data stores cloud applications. *International Journal of Innovative Technology and Exploring Engineering, 8*(9S3). https://doi.org/10.35940/ijitee.I3062.0789S319

Lieponienė, J. (2020). Recent trends in database technology. *Baltic Journal of Modern Computing, 8*(4), 551–559. https://doi.org/10.22364/bjmc.2020.8.4.06

Harezlak, K., Mermon, M., & Kasprowski, P. (2021). A comparison of the efficiency of eye movement data processing with the usage of a multi-model platform. *Procedia Computer Science, 192*, 3070–3078. https://doi.org/10.1016/j.procs.2021.09.079

Pokorný, J. (2019). Integration of relational and graph databases functionally. *Foundations of Computing and Decision Sciences, 44*(4). https://doi.org/10.2478/fcds-2019-0021

Bimonte, S., Gallinucci, E., Marcel, P., & Rizzi, S. (2023). Logical design of multi-model data warehouses. *Knowledge and Information Systems, 65*, 1067–1103. https://doi.org/10.1007/s10115-022-01788-0

Zhang, C., & Lu, J. (2021). Holistic evaluation in multi-model databases benchmarking. *Distributed and Parallel Databases, 39*, 1–33. https://doi.org/10.1007/s10619-019-07279-6

Kazanavičius, J., Mažeika, D., & Kalibatienė, D. (2022). An approach to migrate a monolith database into multi-model polyglot persistence based on microservice architecture: A case study for mainframe database. *Applied Sciences, 12*, 6189. https://doi.org/10.3390/app12126189

Bjeladinovic, S., Marjanovic, Z., & Babarogic, S. (2020). A proposal of architecture for integration and uniform use of hybrid SQL/NoSQL database components. *The Journal of Systems & Software, 164*, 110633. https://doi.org/10.1016/j.jss.2020.110633

Wu, C., Zhu, Q., Zhang, Y., Du, Z., Ye, X., Qin, H., & Zhou, Y. (2017). A NoSQL–SQL hybrid organization and management approach for real-time geospatial data: A case study of public security video surveillance. *ISPRS International Journal of Geo-Information, 6*(1), 21. https://doi.org/10.3390/ijgi6010021

Peña, L. (2023). Holistic approaches to strategically integrating SQL and NoSQL solutions in hybrid architectures for optimized performance and versatile data handling. *Journal of Artificial Intelligence and Machine Learning in Management*, 93. https://doi.org/10.1016/j.jaml.2023.06.004

Krishnapriya, V., Libin, S., & Gibin, G. (2021). A study for integrating SQL and NoSQL databases. *International Conference on Intellectual Property Rights*, 20/02/2021.

Kiehn, F., Schmidt, M., Glake, D., Panse, F., Wingerath, W., Wollmer, B., Poppinga, M., & Ritter, N. (2022). Polyglot data management: State of the art & open challenges*. Proceedings of the VLDB Endowment (VLDB 2022), 15(12), 3750–3753.* https://doi.org/10.14778/3554821.3554891

Zhang, L., Pang, K., Xu, J., & Niu, B. (2022). JSON-based control model for SQL and NoSQL data conversion in hybrid cloud database. *Journal of Cloud Computing*, 11(23). https://doi.org/10.1186/s13677-022-00302-9

Vyawahare, H. R., Karde, P. P., & Thakare, V. M. (2019). Hybrid database model for efficient performance. *Procedia Computer Science*, 152, 172–178. https://doi.org/10.1016/j.procs.2019.05.040

James, B. E., & Asagba, P. O. (2017). Hybrid database system for big data storage and management. *International Journal of Computer Science, Engineering and Applications (IJCSEA)*, 7(3/4), 15. https://doi.org/10.5121/ijcsea.2017.7402

Bjeladinovic, S. (2025). Extending hybrid SQL/NoSQL database by introducing statement rewriting component. *Computer Science and Information Systems*, 00(0), 0000–0000. https://doi.org/10.2298/CSIS123456789X

Ye, F., Sheng, X., Nedjah, N., Sun, J., & Zhang, P. (2023). A benchmark for performance evaluation of a multi-model database vs. polyglot persistence. *Journal of Database Management*, 34(3). https://doi.org/10.4018/JDM.321756

Zdepski, C., Bini, T. A., & Matos, S. N. (2020). New perspectives for NoSQL database design: A systematic review. *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, 68(1), 50–62.

Subramanian, S., & Saravanan, S. (2024). Modern trends in NoSQL databases. International Journal of Computer Trends and Technology, 72(9), 126–130. https://doi.org/10.14445/22312803/IJCTTV72I9P119

Mussbacher, G., Combemale, B., Kienzle, J., Burgueño, L., Garcia-Dominguez, A., et al. (2024). Polyglot software development: Wait, what? IEEE Software, pp. 1–8. https://doi.org/10.1109/MS.2023.3347875

MySQL (2025), MySQL Documentation. Retrieved from https://dev.mysql.com/doc/

MongoDB (2025), MongoDB Documentation. Retrieved from https://www.mongodb.com/docs/

Redis (2025), Redis Documentation. Retrieved from https://redis.io/docs/

The Apache Software Foundation (2025), HBase Documentation. Retrieved from https://hbase.apache.org/book.html

SQLite Consortium, SQLite Documentation. Retrieved from https://www.sqlite.org/docs.html

Oracle Corporation (2025), Oracle Database Documentation. Retrieved from https://docs.oracle.com/en/database/

The PostgreSQL Global Development Group (2025), PostgreSQL Documentation. Retrieved from https://www.postgresql.org/docs/

The Apache Software Foundation (2025), Cassandra Documentation. Retrieved from https://cassandra.apache.org/doc/latest/

Neo4j (2025), Neo4j Documentation Retrieved from https://neo4j.com/docs/

Memcached (2025), Memcached Documentation Retrieved from https://memcached.org/

Microsoft (2025), SQL Server Documentation Retrieved from https://learn.microsoft.com/en-us/sql/sql-server/

Appendix

No	Reference	No	Reference
1	(Majhadi & Machkour, 2021)	16	(Kazanavičius, Mažeika & Kalibatienė, 2022)
2	(Ohlsson & Persson, 2019)	17	(Bjeladinovic, Marjanovic, & Babarogic, 2020)
3	(Kitchenham, 2004)	18	(Wu et al., 2017)
4	(Glake et al., 2022)	19	(Peña, 2023)
5	(Nurhadi et al., 2024)	20	(Krishnapriya, Libin, & Gibin, 2021)
6	(DB-Engines, 2025)	21	(Kiehn et al., 2022)
7	(Khine & Wang, 2019)	22	(Zhang et al., 2022)
8	(Li & Gu, 2019)	23	(Vyawahare, Karde, & Thakare, 2019)
9	(Lajam & Mohammed, 2022)	24	(James & Asagba, 2017)
10	(Kumar & Rampalli, 2019)	25	(Bjeladinovic, 2025)
11	(Lieponienė, 2020)	26	(Ye et al., 2023)
12	(Harezlak, Mermon, & Kasprowski, 2021)	27	(Zdepski, Bini, & Matos, 2020)
13	(Pokorný, 2019)	28	(Subramanian & Saravanan, 2024)
14	(Bimonte et al., 2023)	29	(Mussbacher et al., 2024)
15	(Zhang & Lu, 2021)		